Connectivity-aware Neighborhood Management Protocol in Wireless Sensor Networks

Christoph Weyer, Stefan Unterschütz, and Volker Turau
Institute of Telematics
Hamburg University of Technology, Germany
c.weyer@tu-harburg.de

ABSTRACT

Neighborhood relations are changing over time in wireless sensor networks due to different hardware or environmental effects. These effects and memory limitations require a balanced neighborhood management to ensure agility, stability, symmetry, and connectivity. The proposed neighborhood management protocol Mahalle is optimized with regard to these four criteria. Agility and stability are achieved by ALE, a new adaptive link estimator.

1. INTRODUCTION

Many applications or algorithms for wireless sensor networks depend on a priori knowledge of neighborhood information. This encompasses information about neighborhood relations, i.e., which nodes can communicate with each other, and information about the link quality and reliability between those nodes. Self-stabilizing algorithms are a prominent example of algorithms that highly depend on neighborhood information [3]. The quality of neighborhood information can be described by four criteria: agility, stability, symmetry, and connectivity.

Before accounting a node as a neighbor the quality and reliability of the communication link between the nodes must be assessed. Many parameters of the environment, e.g., multipath effects, lead to a variation of these link properties over time. Due to the characteristics of the wireless channel, neighborhood relations are not known a priori and can only be gathered after the deployment when the network is in operation.

The local view upon the neighborhood of a node forms the topology of the network at the global scope. Variations in hardware, changes in the environment, or node failures lead to dynamic changes of the topology. Therefore, neighborhood relations are not fixed during the whole lifetime of the network. Frequent topology changes lead to a degradation of the algorithms or applications running on top of the formed topology. This aspect of the quality of a neighborhood protocol is captured by the concepts of agility and stability. Agility measures the speed of adapting to desirable events, such as failure of a node respectively a link or deployment of a new nodes. Stability is the ability to ignore transient failures.

Despite variation over time the neighborhood relations must be consistent between nodes. This requirement is expressed by the symmetry property. In particular, if node A is in the neighborhood of B, then node B must also be part of the neighborhood of node A. Otherwise the resulting topology is not useable in most cases. Reason for such an asymmetrical neighborhood relation is the existence of different link qualities caused by different noise levels due to fluctuations in hardware accuracy. Another reason lays in the memory restriction of sensor nodes. Memory restrictions are an important topic in regard to ensuring connectivity. If the density of the network is higher than the number of neighbors that a node can store, a simple neighborhood protocol can lead to disconnected network.

Therefore, a neighborhood management must ensure these four quality criteria. This paper presents Mahalle, a new neighbor management protocol based on adaptive link estimator. Mahalle optimizes the neighborhood relations according to agility, stability, symmetry, and connectivity.

2. RELATED WORK

Research has mainly focused on link estimation so far. Link quality can be assessed by the physical or logical properties. Currently available hardware is providing these physical properties with a high accuracy [2]. The link quality indicator (LQI) or received signal strength indicator (RSSI) are determined by the radio transceiver. However, the values of these metrics are highly hardware specific and must be calibrated for each hardware setting.

Packet reception rate (PRR) is a widely used logical property indicating the link reliability [5]. PRR is calculated from the rate of successfully received periodical broadcast packets. Periodic sending is done by discretizing time into rounds. In each round every node send its broadcast packet. Besides an identifier of the sender these packets contain a sequence number that is used to determine packet loss. In contrast to the physical properties, PRR is the link reliability experienced by applications, but comes at the extra effort of periodical broadcasts. The main drawback of all metrics representing the link behavior is that they exclusively describe a snapshot of the state involving only the recent past. They can not be used as a reliable estimate of the near future, which is needed to provide stability.

Taking the considerations above into account estimating the PRR by evaluating the packet reception rate of periodical broadcasts is the most appropriate way for a solution independent from hardware and environment. In [4] several link

estimation methods are evaluated and compared. The exponentially weighted moving average (EWMA) is one of the most appropriate methods related to fast response, mean error and memory usage. PRR is estimated by PRR $\leftarrow$ PRR $\cdot\, \alpha + (1 - \alpha)$ if a packet was successfully received, and PRR $\leftarrow$ PRR $\cdot\, \alpha$ if a packet loss occurs. A single or consecutive packet loss is detected by a timeout or on the basis of sequence numbers.

The behavior of EWMA is controlled by the parameter α ranging from 0 to 1. For $\alpha = 0.99$ EWMA is called stable and has a high accuracy with a low mean square error but with a high crossing time. The metric crossing time is defined by the time the estimator needs after detecting a new link to come up with a PRR value having an error of at most ϵ. In the following ϵ is assumed to be 0.05. For $\alpha = 0.915$ EWMA is called agile. This value leads to a much shorter crossing time and a smaller reaction time to link changes, but suffers from a high variation with a high mean square error. In [4] the stable EWMA is preferred to the agile in order to produce more stable link estimations over time. On top of such a link estimator in [5] a neighborhood protocol that is aware of asymmetrical links is described. The realization of this, especially with a neighbor table size of 40 entries, is not further explained.

TinyOS 2.x uses the Link Estimation Exchange Protocol (LEEP) to estimate link reliability and manage neighborhood information [1]. The Extra Expected number of Transmissions (EETX) is used as a link-quality metric. EETX is the expected value of a geometrically distribution with the probability PRR not counting the successful transmission. Therefore, EETX is defined as EETX $= (1/\text{PRR}) - 1$. The bidirectional EETX value is based on the product of inbound and outbound packet reception rate that are discrete recursively estimated. Inbound PRR is estimated as the reception rate at node A by receiving packets from node B. The reception rate from node A at node B is the outbound PRR of node A. The neighborhood table size is limited to 10 neighbors in TinyOS. To share the outbound quality information the complete neighborhood table must be exchanged. If the complete link information does not fit into one packet, a round-robin schedule is established. If the neighborhood table is full and a new node is detected, the neighbor with a low EETX value if existent is evicted. This policy is simple, but can lead to disconnected topologies in dense networks.

3. ADAPTIVE LINK ESTIMATOR

Since the proposed adaptive link reliability estimator can be used independently from the neighborhood management protocol, it is specified separately in this section. As described in [4] the PRR is estimated by observing the successful delivery of periodical broadcast packets. If application-specific packet rate is sufficient, the estimator information can be sent piggybacked in the data packets in order to reduce the periodical broadcast packets. EWMA in agile mode is a fast responding link estimator while in stable mode a low mean square error is achieved. The main goal of the adaptive link estimator proposed in this paper is to combine the strength of both modes.

The adaptive link estimator (ALE) uses EWMA as a basis. It has the capability to adjust the parameter α depending on the current estimated link quality. To improve the crossing time the initial PRR is set to the threshold value PRR_{bad}, whenever the first packet is received from a node. In this paper PRR_{bad} is set to 0.5. Additionally, when the PRR is below PRR_{bad} or a new link must be estimated the agile mode is used in order to estimate quickly the real link quality. ALE switches from agile into stable mode after 30 rounds in case that the PRR is greater than PRR_{bad} during this period for a more accurate estimation. This mechanism is called raising mechanism, which improves the crossing time of ALE compared to EWMA. In Figure 1 the crossing time of the different estimators is compared. The crossing time of the stable EWMA is very high compared to the agile mode. Due to the threshold PRR_{bad} and the raising mechanism the crossing time of ALE is even faster than the agile EWMA, especially for good links. Observe that the crossing time of ALE is smaller than 30 for all high values of PRR. This justifies the choice made above for the number of rounds to switch from the agile to stable mode.

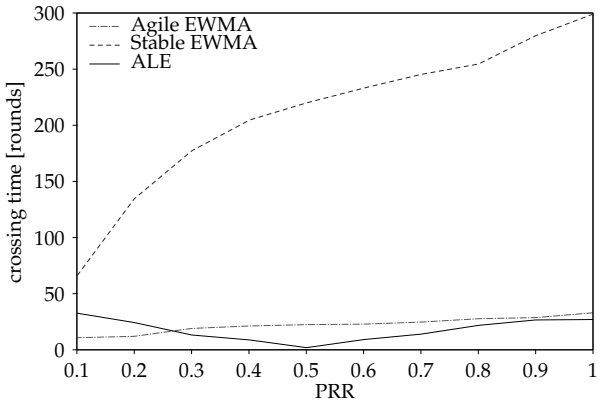
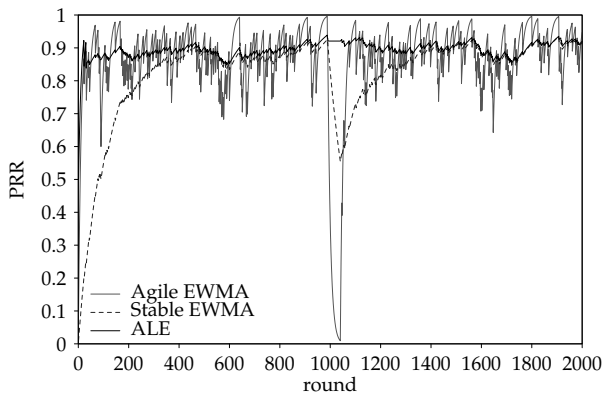


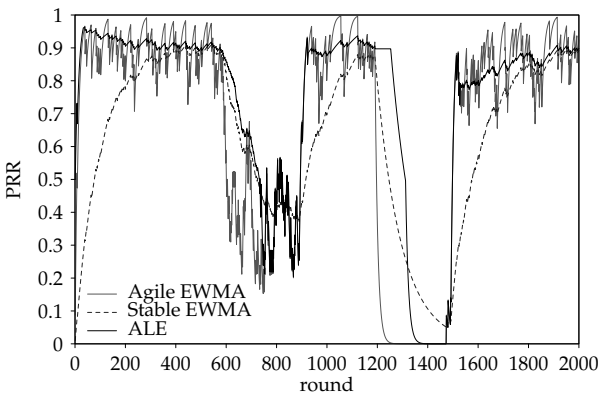
Figure 1: Crossing Time

Another mechanism that improves ALE is the so called dropping mechanism. With this mechanism a good link is trusted even if currently no communication is possible. Doing so the stability of the resulting network topology is increased. Whenever a good link, with PRR $>$ PRR_{good} suffers from consecutive packet loss, the PRR is kept constant for 60 rounds. This number is a parameter of ALE and specifies the time that is conceded to a node to recover from a transient failure.

In Figure 2 the behavior of EWMA and ALE is compared over time. As shown in both figures, ALE outperforms the agile mode of EWMA in terms of mean square error. Figure 2 (a) shows a link with a fixed real PRR of 0.89 and a short link interruption of 50 packets in round 1000. This figure depicts the improvements during startup of ALE and especially during the short interruption compared to stable EWMA. Figure 2 (b) shows a link with a PRR of 0.89, which is dropping during round 600 and 900 to a PRR of 0.4 and is interrupted during round 1200 and 1500. Here the improvements introduced by the dropping mechanism are depicted. If the PRR is dropping below PRR_{bad} the estimation mode of ALE switches back to the agile mode in order to react faster on link changes.



(a) Real PRR = 0.89 and short message lost



(b) Variation of PRR over time

Figure 2: Comparison between EWMA and ALE

Normally a link is considered as a neighbor if the link quality is above a given threshold, here defined by PRR_{good} and is often set to 0.8. In contrast, a neighbor is disregarded if its quality drops below PRR_{good}. Doing so oscillation effects can occur. Therefore, a hysteresis is used to decrease oscillation effects as much as possible. Several simulations revealed that a range of ± 0.06 is a good choice between oscillation and size of the hysteresis. This leads to $\text{PRR}_{in} = 0.86$ and $\text{PRR}_{out} = 0.74$.

4. NEIGHBORHOOD MANAGEMENT

The neighborhood management protocol Mahalle uses the information about link qualities between nodes provided by the adaptive link estimator described in the previous section. ALE delivers only information about one direction of a communication link. If node A receives packets from node B it can calculate the PRR of node B at node A by using ALE. But this information is of no significance for the PRR of node A at node B. To ensure symmetry of the communication links between neighbors the PRR values of both directions of the link must be exchanged.

The PRR is normally exchanged between nearby nodes by using periodical broadcasts. In general, three different approaches exist. The simplest way is to put the estimated PRR values of all known nodes in each packet. The main drawback of this approach is the required size of such a packet. Due to the high bit error rate, the probability of packet loss is increased, which influences the PRR estimation. The second approach is to send only the PRR values that have changed lately. This decreases the amount of information sent in one packet, but the size varies. If a packet gets lost, the information must be resent even if the PRR is unchanged. The complexity of this approach is thereby increased. Another approach is to send only fractions of the PRR values in a round-robin schedule. Doing so, all packets are equally sized and every PRR value is broadcasted with a fixed schedule. Mahalle uses the last approach to exchange link information between neighboring nodes. One additional benefit of this is that additional information can be included into the broadcast packets, e.g., the node state needed by self-stabilizing algorithms [3].

The node table consists of entries to store the neighbor list and the preparation list. The neighbor list contains the nodes that are treated as neighbors. All other entries in the node table are used by the preparation list, where potential neighbors are stored in order to be able to estimate their link quality. At startup the whole node table is occupied by the preparation list. If a node is evicted from the node table, it is inserted into a blacklist for some time in order to prevent them to reenter immediately. The number of entries in the node table (N_t) and the ratio between the maximum number of entries in the neighbor list (N_n) and the preparation list (N_p) are compile-time parameters of Mahalle and correlate to the expected average network density.

In every round each node broadcasts a neighborhood packet. This packet contains the identifier of the node, a sequence number, and one entry of the node table. This information contains of the identifier of the neighbor and the estimated PRR. The round-robin schedule consists of N_t rounds, so that each entry of the node table is rebroadcasted in every N_t'th round. This information is used by the nodes that are receiving the packets.

Node table entries contain of the following information: a flag that indicates if the node is in the neighbor list, estimated inbound PRR, the received outbound PRR, the number of neighbors and overlapping neighbors, a flag indicating symmetry, and the age of the entry. Whenever a node receives a broadcast from a node that is contained in its node table, the corresponding entry is updated. Each packet is used to estimate the inbound PRR. When the packet contains the identifier of the receiving node, the outbound PRR in the packet is stored in the table. From all packets during N_t rounds the node can also identify the number of neighbors of the sending node and the number of overlapping neighbors, the nodes that are common for both.

When a packet is received by an unknown node and the node table has a free entry, the node is inserted into the preparation list (if the node table is full the packet is ignored). If the estimated PRR of the inserted node is above PRR_{in} the

node is a candidate to be placed into the neighbor list. If the node stays longer than 50 rounds in the preparation list without entering the neighbor list, the node is deleted from the preparation list and added to the blacklist. If a node is in the neighbor list and its PRR is below PRR_{out} the node is removed from the list and added to the blacklist.

If the neighbor list is full and in the preparation list a link with PRR > 0.86 exists an entry from the neighbor list must be evicted to provide space for the new potential neighbor. Therefore, a screening process exists to choose the most appropriate neighbor that is evicted from the neighbor list. The following rules are applied to the list containing all neighbors and the candidates from the preparation list. The rules are processed one-by-one until a single node remains. If the new potential neighbor is selected the neighbor list is kept unchanged and the node is inserted in the blacklist and removed from the preparation list.

Rule 1: Select asymmetry links

Rule 2: Select nodes with neighbors

Rule 3: Select highest overlapping

Rule 4: Select less unknown neighbors

Rule 5: Select lowest bidirectional PRR

Rule 6: Select newest neighbors

The first rule tries to evict nodes that are asymmetric from the list. If several nodes are found or none of the entries are asymmetrical, the next rule is applied. This rule protects those nodes from being removed that have no other neighbor. The next rule selects those nodes that have the highest number of overlapping neighbors, since those nodes are connected via one of the common neighbors with a high probability. If there are still several nodes having the same properties those nodes are selected, which provide the smallest number of unknown neighbors. Those nodes are not that important for insuring connectivity. These last two rules are protected by a hysteresis in order to prevent an oscillation effect. In rule 5 the nodes with the lowest bidirectional PRR are selected to be removed from the neighbor table. This PRR is the product of the estimated inbound and received outbound PRR. If still more than one node is selected the newest neighbors are preferred, since the older neighbors insure more stability. If still several nodes are in the list after applying the last rule, a node is picked randomly.

The ns-2 simulator is used for validating Mahalle. To compare the connectivity awareness of Mahalle a simple neighborhood management protocol is also implemented, which uses only the inbound and outbound PRR values provided by ALE. Additional the LEEP [1] protocol integrated in TinyOS 2.x (see Section 2) is ported to ns-2. Several different densities with 200 randomly placed nodes are investigated. For each density 50 different topologies are used. The used propagation model provides around 15% of asymmetrical links, which means that the inbound and outbound PRR differs more than 0.15. In Figure 3 the results for a density of 24 is shown. 180 Nodes are started at round 0 and 20 nodes where added after 1750 rounds. The connectivity

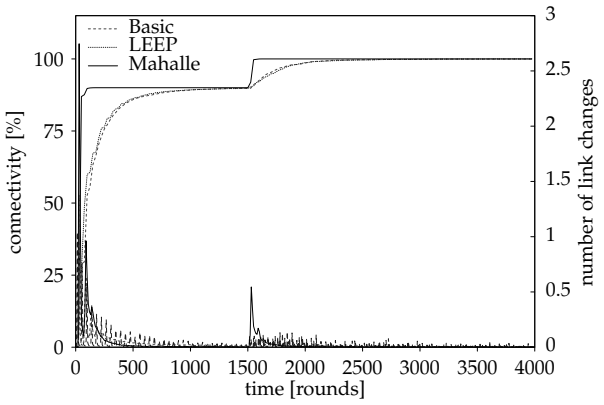


Figure 3: Comparison with network density 24

and the number of link changes are shown over time. Mahalle provides very fast neighborhood detection and reaches 100% connectivity. Due to the large density of 24 potential neighbors, LEEP and the basic protocol are providing only a slow convergence. Connectivity is not fully reached by these two approaches after adding nodes. The number of link changes during the stable phases is also decreased in Mahalle.

5. CONCLUSION

The presented neighborhood management protocol Mahalle and the underlying adaptive link estimator have shown good results in terms of agility, stability, symmetry, and stability. Mahalle outperforms in terms of stabilization speed. The next steps will be porting Mahalle to TinyOS and using it in a real long-term deployment at our campus.

6. ACKNOWLEDGEMENT

We would like to thank Ting-Fu Chen for his help in developing a preliminary version of Mahalle.

7. REFERENCES

[1] O. Gnawali. *TinyOS Extension Proposal (TEP) 124: The Link Estimation Exchange Protocol (LEEP)*, Feb. 2007. http://www.tinyos.net/tinyos-2.x/doc/pdf/tep124.pdf.

[2] K. Srinivasan and P. Levis. RSSI is Under Appreciated. In *Proceedings of the Third Workshop on Embedded Networked Sensors (EmNets'06)*, Harvard University Cambridge, MA, USA, May 2006.

[3] V. Turau and C. Weyer. Randomized Self-stabilizing Algorithms for Wireless Sensor. In *Proceedings of the International Workshop on Self-Organizing Systems (IWSOS'06)*, Passau, Germany, Sept. 2006.

[4] A. Woo and D. Culler. Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks. Technical Report UCB/CSD-03-1270, U.C. Berkeley Computer Science Division, Sept. 2003.

[5] A. Woo, T. Tong, and D. Culler. Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, USA, Nov. 2003.